УДК 004.428.4 DOI: 10.18698/2541-8009-2020-02-583

ПРОГРЕССИВНЫЕ ВЕБ-ПРИЛОЖЕНИЯ: ОБЪЕДИНЯЮЩАЯ ТЕХНОЛОГИЯ ДЛЯ ВЕБ- И НАТИВНЫХ ПРИЛОЖЕНИЙ

П.В. Киселев

kiselevpv@student.bmstu.ru
SPIN-код: 6338-7026

МГТУ им. Н.Э. Баумана, Москва, Российская Федерация

Аннотация

Рассмотрены прогрессивные веб-приложения, которые могут служить в качестве объединяющей технологии для веб- и нативных приложений. Сначала функции представлены для ознакомления, затем тщательно исследована их производительность. Для сравнения разработаны гибридное, интерпретируемое и прогрессивное веб-приложения. Тестирование производительности осуществлялось по следующим параметрам: объем памяти, занимаемой приложением на устройстве; скорость инициализации приложения; время, прошедшее от запуска приложения до полной отрисовки компонента с заголовком приложения. По итогам тестирования выявлено, что прогрессивные веб-приложения могут стать унификатором для нативной веб-разработки без использования кроссплатформенных сред.

Ключевые слова

Прогрессивные веб-приложения, Service Worker, кроссплатформенность, кроспратформенная разработка, мобильный интернет, мобильная платформа, интерфейс, гибридный подход, интерпретируемый подход

Поступила в редакцию 20.01.2020

Введение. При разработке мобильных приложений возможность использовать один и тот же код для передачи данных между нативными приложениями, веб- и мобильными платформами изначально отсутствовала. Это было связано с несовместимостью кода нативных приложений для разных операционных систем и требовало создания отдельных проектов и использования иных сред разработки в случаях, когда было необходимо организовать поддержку нескольких мобильных платформ [1–3].

Для компаний, у которых нет ресурсов или желания нанимать специализированных разработчиков мобильных приложений для каждой целевой платформы, кроссплатформенная разработка стала популярной альтернативой [4]. Как правило, это не только облегчает разработку, но и позволяет быстрее выходить на рынок [5]. Хотя ее применение и репутация варьируются в научных кругах и отраслевых сообществах [6, 7], идея единой базы кода, которая может быть развернута на нескольких платформах, считается привлекательной по ряду причин, к которым относятся бюджет, человеческие ресурсы и имеющиеся знания [8].

В кроссплатформенной разработке существует несколько подходов [9]. Также доступно множество фреймворков с открытым исходным кодом либо в виде платных проприетарных продуктов, которые могут быть использованы при разработке. Примерами популярных фреймворков являются Ionic Framework, Apache Cordova, React Native и Xamarin [10]. Эти фреймворки представляют три технологически разных подхода.

Ionic Framework и Apache Cordova принадлежат к *гибридному подходу*, в нем компоненты пользовательского интерфейса структурированы и стилизованы исключительно с использованием веб-технологий, включая HTML и CSS. React Native в *интерпретируемом подходе* не зависит от веб-представления, поскольку вместо этого применяется интерпретатор JavaScript вместе с нативными мостами, что позволяет использовать компоненты нативного интерфейса вместо веб-компонентов HTML. Xamarin обычно относят к *подходу кросс-компиляции*, поскольку он компилирует код C# в нативные бинарные файлы для каждой поддерживаемой платформы. Это позволяет создавать нативные приложения, которые не зависят от интерпретаторов или веб-представления [11].

До недавнего времени веб-платформа отставала в ориентированности на мобильные устройства с точки зрения возможности конкуренции с нативными или кроссплатформенными приложениями [12]. Новый набор стандартов, отстаиваемый группой Google Web Fundamentals, стремится восполнить этот пробел, внедрив в веб-приложения такие функции, как поддержка автономности, фоновая синхронизация и возможность установки приложений на домашний экран. Этот подход известен под названием «Прогрессивные веб-приложения» (Progressive Web Applications — PWA). Это попытка унификации мобильных приложений, где веб-приложения можно устанавливать и распространять без размещения в магазинах приложений, работать без подключения к Интернету, получать push-уведомления, при этом PWA имеют вид обычных приложений. Это возможно благодаря сервису Service Worker API, который дает разработчикам возможность использовать фоновый скрипт, выступающий в роли прокси-сервера между сетью и устройством.

Несмотря на то что индустрия проявляет интерес к прогрессивным веб-приложениям, в этой области представлено очень мало научных исследований. Следовательно, существует значительный исследовательский потенциал. Цель этой статьи — дать представление о концепциях и технологиях, лежащих в основе прогрессивных веб-приложений. Кроме того, статья должна продемонстрировать особенности PWA и обеспечить техническое сравнение. Для сравнения скорости инициализации, размера (объема занимаемой приложением памяти) и скорости рендеринга приложений было разработано три мобильных приложения с использованием гибридного, интерпретируемого и PWA-подходов.

Разработка приложений. Для сравнения были разработаны три мобильных приложения, позволяющие лучше понять возможности прогрессивных веб-

приложений. Гибридное приложение было разработано с использованием технологии Ionic Framework, интерпретируемое приложение — React Native, а прогрессивное веб-приложение — React.js.

В приложениях используется шаблон навигации master-detail, где в master-представлении отображается список интерактивных элементов, получаемый из API, а при нажатии на элемент приложение переходит к подробному виду, отображая изображение элемента вместе с его автором и заголовком. Данные загружаются с сайта www.reddit.com/r/Pictures/.json. Все три мобильных приложения имеют открытый исходный код и доступны для проверки результатов [13].

Тестирование производительности. Тесты проводили на устройстве Google Nexus 6P под управлением операционной системы Android версии 8.0. Целью было собрать данные о различиях в производительности между разными реализациями приложений, которые обсуждались выше. Тест производительности предусматривал контроль следующих параметров: объем памяти, занимаемой приложением на устройстве; скорость инициализации приложения; время, прошедшее от запуска приложения до полной отрисовки компонента с заголовком приложения.

Объем занимаемой приложением памяти. Информация о размере приложения находится на устройстве Android по следующему адресу: Настройки / Приложения / [имя приложения].

Скорость инициализации приложения. Параметр измеряли с помощью отладочного моста Android (ADB), за это отвечает команда am_activity_launch_time, которая измеряет время, затраченное на инициализацию приложения. Тест проводили 10 раз для каждого приложения, поскольку при каждом запуске результаты различались. В качестве результата представлено среднее время.

Время, прошедшее от запуска приложения до полной отрисовки компонента с заголовком приложения. Поскольку в отладочном мосту Android не было найдено подходящей команды, параметр измеряли следующим образом: с помощью онлайн-секундомера начинали отсчет с момента нажатия на иконку приложения на домашнем экране до момента отрисовки компонента с заголовком. В разработанных приложениях этот компонент отображается первым, поэтому он и был выбран для измерения времени отрисовки. Так как измерение основывалось на реакции человека, которая, в свою очередь, имеет погрешность, тест был проведен 10 раз для каждого приложения. В качестве результата представлено среднее время.

Сравнение функций. Покажем различия между интерпретируемыми, гибридными, нативными и прогрессивными веб-приложениями, сравнив набор их основных функций и концепций (табл. 1). В примечаниях к таблице также приведены некоторые технические сведения о фреймворках.

Таблица 1

Сравнение особенностей подходов

Характеристика	Интерпретируемое	PWA	Гибридное	Нативное
Возможность установки	Да	Да	Да	Да
Доступно автономно	Да	Да	Да	Да
Тестируемо до установки	Нет	Да	Нет	Нет
Может быть размещено в магазине приложений	Да	Да[а]	Да	Да
Push-уведомления	Да	Да[б]	Да	Да
Кроссплатформенность	Да	Есть ограничения[в]	Да	Нет
Доступ к аппаратным средствам и API платформы	Да	Есть ограничения[г]	Да[д]	Да
Фоновая синхронизация	Да	Да	Да	Да

Примечания:
а) прогрессивные веб-приложения доступны в Google Play и Microsoft Store; б) имеется поддержка push-уведомлений через Push API, но она ограничена определенными браузерами [14]; в) ограничения в основном касаются iOS. Приложение может хранить только до 50 Мб данных на устройстве. Нет доступа к некоторым функциям, таким как Bluetooth, Touch ID, идентификатор лица, ARKit, датчик высотомера, информация о батарее. Нет поддержки фоновой синхронизации и push-уведомлений, нет возможности разместить банер на сайте, чтобы предложить пользователю установить приложение как PWA, но можно использовать нативный интерфейс браузера для установки; г) PWA может использовать API-интерфейсы HTML5 для доступа к аппаратным средствам и платформе в дополнение к функциям и возможностям, предоставляемым Service Worker; д) доступ к аппаратным средствам и платформе в гибридных приложениях обычно предоставляется с помощью Apache Cordova — библиотеки для перехода от компонентов веб-представления к компонентам нативного приложения и API-интерфейсам устройства.

Технологии и концепции. Для веб-разработки существует множество фреймворков [15]. Группа Web Fundamentals предоставила возможность не зависеть от фреймворков, внедрив поддержку PWA в трех разных фреймворках [16].

Service Worker отвечает за большинство основных функций, связанных с прогрессивными веб-приложениями [17]. PWA не может должным образом работать в браузерах без поддержки Service Worker. Service Worker регистрируется при первом посещении пользователя. Он состоит из файла JavaScript, содержащего хуки жизненного цикла для бизнес-логики и управления кешем. Он может использоваться для обработки таких задач, как фоновая синхронизация [18],

механизмы кэширования данных и оболочки приложения, а также перехвата сетевых запросов [19].

Оболочка приложения определяется группой Google Web Fundamentals как «минимально необходимая связка HTML, CSS и JavaScript, обеспечивающая пользовательский интерфейс» [19]. В документации перечисляются три критерия оболочки: быстрое время загрузки, кэширование и отображение динамического содержимого. Данные получаются из внешних API.

Для предоставления возможности разработчикам приложений изменять некоторые настройки используется файл манифеста. Эти параметры включают путь к изображению логотипа, имя приложения, заставку и многое другое. Таким образом, манифест можно использовать для изменения поведения и стилизации приложений PWA.

HTTPS требуется для Service Worker из соображений безопасности, так как после регистрации в браузере Service Worker реагирует на все события [17]. Принудительное использование HTTPS обязательно, поскольку Service Worker может перехватывать соединения и фильтровать ответы [17].

Сравнение трех разных параметров: объема занимаемой приложением памяти, время инициализации и время рендеринга приложения представлено в табл. 2. PWA показала разные результаты, когда браузер Chrome не работал в фоновом режиме и когда он работал в фоновом режиме независимо от открытого веб-сайта. Это связано с тем, что PWA использует браузер.

Таблица 2

Сравнительный анализ разработанных приложений

Параметр	Гибридное	Интерпретируемое	PWA
Объем занимаемой памяти, Мб	4,54 Мб	16,44	106 Кб
Время инициализации, мс	855	240	228
Время рендеринга, мс	9188,1	854	3148[а] 1294[б]
Примечания: а) Google Chrome не работал в фоновом режиме; б) Google Chrome работал в фоновом режиме (независимо от открытого веб-сайта).			

Заключение. Индустрия инвестирует ресурсы в прогрессивные веб-приложения (PWA) и разработку учебных материалов. Разработанные приложения доступны онлайн и имеют открытый исходный код для дальнейших исследований и проверки результатов, представленных в статье.

По состоянию на июль 2019 г. группа Google Web Fundamentals стала одной из ведущих движущих сил, пропагандирующих PWA. Она считается основным издателем обучающих материалов.

Текущее состояние прогрессивных веб-приложений обусловлено отсутствием доступа к некоторым аппаратным и платформенным API-интерфейсам, к которым могут обращаться только кроссплатформенные и нативные приложения.

У PWA есть большой потенциал для того, чтобы стать унификатором для нативной веб-разработки без использования кроссплатформенных сред. Для конечного пользователя процесс установки PWA становится все более похожим на установку обычного приложения. Веб-приложения могут выглядеть и работать как нативные, гибридные и интерпретируемые приложения. Хотя существуют некоторые ограничения доступа к аппаратным и платформенным API-интерфейсам, которых нет в других подходах, требования к продукту и спецификации в итоге станут решающим фактором при выборе подхода для разработки.

Литература

[1] Perchat J., Desertot M., Lecomte S. Component based framework to create mobile cross-platform applications. *Procedia Comput. Sci*., 2013, vol. 19, pp. 1004–1011 DOI: https://doi.org/ 10.1016/j.procs.2013.06.140

[2] Heitkotter H., Majchrzak T.A., Kuchen H. Cross-platform model-driven development of mobile applications with md^2. *Proc. SAC'13*, 2013, pp. 526–533. DOI: https://doi.org/ 10.1145/2480362.2480464

[3] Majchrzak T.A., Heitkotter H. Status quo and best practices of app development in regional companies. *Proc. WEBIST*, 2013, pp. 189–206 2014. DOI: https://doi.org/ 10.1007/978-3-662-44300-2_12

[4] Malavolta I., Ruberto S., Soru T., et al. Hybrid mobile apps in the Google Play Store: an exploratory investigation. *Proc. 2nd ACM Int. Conf. Mobile Software Engineering and Systems*, 2015, pp. 56–59. DOI: https://doi.org/10.1109/MobileSoft.2015.15

[5] Rahul R., Tolety S.B. A study on approaches to build cross-platform mobile applications and criteria to select appropriate approach. *Proc. INDICON*, 2012, pp. 625–629. DOI: https://doi.org/10.1109/INDCON.2012.6420693

[6] Puvvala A., Dutta A., Roy R., et al. Mobile application developers' platform choice model. *Proc. 49th HICSS*, 2016, pp. 5721–5730. DOI: https://ieeexplore.ieee.org/__document/7427896

[7] Mercado I.T., Munaiah N., Meneely A. The impact of cross-platform development approaches for mobile applications from the user's perspective. *Proc. WAMA*, 2016, pp. 43–49. DOI: https://doi.org/10.1145/2993259.2993268

[8] Corral L., Janes A., Remencius T. Potential advantages and disadvantages of multiplatform development frameworks–a vision on mobile environments. *Procedia Comput. Sci*., 2012, vol. 10, pp. 1202–1207. DOI: https://doi.org/10.1016/j.procs.2012.06.173

[9] Heitkotter H., Hanschke S., Majchrzak T.A. Evaluating cross-platform development approaches for mobile applications. *Proc. 13th WEBIST*, 2017, pp. 344–351. URL: https://www.scitepress.org/Papers/2017/63537/63537.pdf (дата обращения: 15.10.2019).

[10] Majchrzak T.A., Biørn-Hansen A., Grønli T.-M. Comprehensive analysis of innovative crossplatform app development frameworks. *Proc. HICSS*, 2017. DOI: https://doi.org/ 10.24251/ HICSS.2017.745

[11] Latif M., Lakhrissi Y., Nfaoui E.H., et al. Cross platform approach for mobile application development: a survey. *PROC. IT4OD*, 2016. DOI: https://doi.org/10.1109/IT4OD.2016.7479278
[12] Puder A., Tillmann N., Moskal M. Exposing native device APIs to web apps. *Proc. MOBILESoft*, 2014, pp. 18–26. DOI: https://doi.org/10.1145/2593902.2593908
[13] GitHub: веб-сайт. URL: https://github.com/Cappy/PWA (дата обращения: 10.12.2019).
[14] W3C – Push API. *w3.org: веб-сайт*. URL: https://www.w3.org/TR/push-api/ (дата обращения: 01.11.2019).
[15] Knox C.M. Get some latest 2019 trends for cross-platform mobile app development. *medium.com: веб-сайт*. URL: https://medium.com/@michael_tech/get-some-latest-2019-trends-for-cross-platform-mobile-app-development-e5dae65115fa (дата обращения: 04.11.2019).
[16] Osmani A. Getting started with progressive web apps. *developers.google.com: веб-сайт*. URL: https://developers.google.com/web/updates/2015/12/getting-started-pwa (дата обращения: 01.11.2019).
[17] Gaunt M. Service Workers: an introduction. *developers.google.com: веб-сайт*. URL: https://developers.google.com/web/fundamentals/primers/service-workers/ (дата обращения: 04.11.2019).
[18] Archibald J. Instant loading: building offline-first progressive web apps – Google I/O 2016. *developers.google.com: веб-сайт*. URL: https://developers.google.com/web/shows/google-io/2016/instant-loading-building-offline-first-progressive-web-apps-google-io-2016 (дата обращения: 04.11.2019).
[19] Osmani A., Gaunt M. Instant loading web apps with an application shell architecture. *developers.google.com: веб-сайт*. URL: https://developers.google.com/web/updates/2015/11/app-shell (дата обращения: 04.11.2019).

Киселев Павел Владимирович — магистрант кафедры «Программное обеспечение ЭВМ и информационные технологии», МГТУ им. Н.Э. Баумана, Москва, Российская Федерация.

Научный руководитель — Григорьев Александр Сергеевич, кандидат технических наук, доцент кафедры «Программное обеспечение ЭВМ и информационные технологии», МГТУ им. Н.Э. Баумана, Москва, Российская Федерация.

Ссылку на эту статью просим оформлять следующим образом:
Киселев П.В. Прогрессивные веб-приложения: объединяющая технология для веб- и нативных приложений. *Политехнический молодежный журнал*, 2020, № 02(43). http://dx.doi.org/10.18698/2541-8009-2020-02-583

PROGRESSIVE WEB APPLICATIONS: COMBINING TECHNOLOGY FOR WEB AND NATIVE APPLICATIONS

P.V. Kiselev

kiselevpv@student.bmstu.ru
SPIN-code: 6338-7026

Bauman Moscow State Technical University, Moscow, Russian Federation

Abstract

The paper discusses progressive web applications that can serve as a unifying technology for web and native applications. First, the functions are presented for review, then their performance is carefully examined. For comparison, hybrid, interpretable, and progressive web applications have been developed. Performance testing was carried out according to the following parameters: the amount of memory occupied on the device by the application; application initialization speed; the time elapsed from launching the application to fully rendering the component with the application title. The results of testing revealed that progressive web applications can become a unifier for native web development without the use of cross-platform environments.

Keywords

Progressive web applications, Service Worker, cross-platform, cross-platform development, mobile Internet, mobile platform, interface, hybrid approach, interpreted approach

Received 20.01.2020

References

[1] Perchat J., Desertot M., Lecomte S. Component based framework to create mobile cross-platform applications. *Procedia Comput. Sci.*, 2013, vol. 19, pp. 1004–1011 DOI: https://doi.org/ 10.1016/ j.procs.2013.06.140

[2] Heitkotter H., Majchrzak T.A., Kuchen H. Cross-platform model-driven development of mobile applications with md^2. *Proc. SAC'13*, 2013, pp. 526–533. DOI: https://doi.org/ 10.1145/2480362.2480464

[3] Majchrzak T.A., Heitkotter H. Status quo and best practices of app development in regional companies. *Proc. WEBIST*, 2013, pp. 189–206 2014. DOI: https://doi.org/ 10.1007/978-3-662-44300-2_12

[4] Malavolta I., Ruberto S., Soru T., et al. Hybrid mobile apps in the Google Play Store: an exploratory investigation. *Proc. 2nd ACM Int. Conf. Mobile Software Engineering and Systems*, 2015, pp. 56–59. DOI: https://doi.org/10.1109/MobileSoft.2015.15

[5] Rahul R., Tolety S.B. A study on approaches to build cross-platform mobile applications and criteria to select appropriate approach. *Proc. INDICON*, 2012, pp. 625–629. DOI: https://doi.org/10.1109/INDCON.2012.6420693

[6] Puvvala A., Dutta A., Roy R., et al. Mobile application developers' platform choice model. *Proc. 49th HICSS*, 2016, pp. 5721–5730. DOI: https://ieeexplore.ieee.org/ document/7427896

[7] Mercado I.T., Munaiah N., Meneely A. The impact of cross-platform development approaches for mobile applications from the user's perspective. *Proc. WAMA*, 2016, pp. 43–49. DOI: https://doi.org/10.1145/2993259.2993268

[8] Corral L., Janes A., Remencius T. Potential advantages and disadvantages of multiplatform development frameworks–a vision on mobile environments. *Procedia Comput. Sci.*, 2012, vol. 10, pp. 1202–1207. DOI: https://doi.org/10.1016/j.procs.2012.06.173

[9] Heitkotter H., Hanschke S., Majchrzak T.A. Evaluating cross-platform development approaches for mobile applications. *Proc. 13th WEBIST*, 2017, pp. 344–351. URL: https://www.scitepress.org/Papers/2017/63537/63537.pdf (accessed: 15.10.2019).

[10] Majchrzak T.A., Biørn-Hansen A., Grønli T.-M. Comprehensive analysis of innovative crossplatform app development frameworks. *Proc. HICSS*, 2017. DOI: https://doi.org/10.24251/HICSS.2017.745

[11] Latif M., Lakhrissi Y., Nfaoui E.H., et al. Cross platform approach for mobile application development: a survey. *PROC. IT4OD*, 2016. DOI: https://doi.org/ 10.1109/IT4OD.2016.7479278

[12] Puder A., Tillmann N., Moskal M. Exposing native device APIs to web apps. *Proc. MOBILESoft*, 2014, pp. 18–26. DOI: https://doi.org/10.1145/2593902.2593908

[13] GitHub: website. URL: https://github.com/Cappy/PWA (accessed: 10.12.2019).

[14] W3C – Push API. *w3.org: website*. URL: https://www.w3.org/TR/push-api/ (accessed: 01.11.2019).

[15] Knox C.M. Get some latest 2019 trends for cross-platform mobile app development. medium.com: website. URL: https://medium.com/@michael_tech/get-some-latest-2019-trends-for-cross-platform-mobile-app-development-e5dae65115fa (accessed: 04.11.2019).

[16] Osmani A. Getting started with progressive web apps. *developers.google.com: website*. URL: https://developers.google.com/web/updates/2015/12/getting-started-pwa (accessed: 01.11.2019).

[17] Gaunt M. Service Workers: an introduction. *developers.google.com: website*. URL: https://developers.google.com/web/fundamentals/primers/service-workers/ (accessed: 04.11.2019).

[18] Archibald J. Instant loading: building offline-first progressive web apps – Google I/O 2016. *developers.google.com: website*. URL: https://developers.google.com/web/shows/google-io/2016/instant-loading-building-offline-first-progressive-web-apps-google-io-2016 (accessed: 04.11.2019).

[19] Osmani A., Gaunt M. Instant loading web apps with an application shell architecture. *developers.google.com: website*. URL: https://developers.google.com/web/updates/2015/11/app-shell (accessed: 04.11.2019).

Kiselev P.V. — Master's Degree Student, Department of Computer Software and Information Technology, Bauman Moscow State Technical University, Moscow, Russian Federation.

Scientific advisor — Grigoriev A.S., Cand. Sc. (Eng.), Assoc. Professor, Department of Computer Software and Information Technology, Bauman Moscow State Technical University, Moscow, Russian Federation.

Please cite this article in English as:
Kiselev P.V. Progressive web applications: combining technology for web and native applications. *Politekhnicheskiy molodezhnyy zhurnal* [Politechnical student journal], 2020, no. 02(43). http://dx.doi.org/ 10.18698/2541-8009-2020-01-583.html (in Russ.).